

Search: ☒ The Guide ☒ The ACM Digital Library

[+identifying](#) [+string](#) [+substitute](#) [+replace](#) [+macro](#) [+transform](#) [+define](#)

THE ACM DIGITAL LIBRARY

 [Feedba](#)

Terms used [identifying](#) [string](#) [substitute](#) [replace](#) [macro](#) [transform](#) [define](#)

Sort results
by

relevance



Display results

expanded form



☒ Save results to a Binder

Ti

☒ Search Tips

Ti

☐ Open results in a new window


Results 1 - 20 of 94

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) ne:

1 Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 Proceedings of the 1997 conference of the Centre for Advanced S

Full text available:  pdf(4.21 MB)

Additional Information: full citation, abstract, ref rence

Understanding distributed applications is a tedious and difficult task. Visualiza often used to obtain a better understanding of the execution of the application event tracer developed at the University of Waterloo. However, these diagram provide the user with the desired overview of the application. In our experienc of non-trivial commun ...

2 The Vienna Definition Language

Peter Wegner


January 1972 ACM Computing Surveys (CSUR), Volume 4 Issue 1

Full text available:  pdf(3.89 MB) Additional Information: full citation, references, citings, index terms

3 Optimization of functional programs by grammar thinning

Adam Webber

March 1995 ACM Transactions on Programming Languages and Systems (TOPLA)

Full text available:  pdf(2.43 MB)

Additional Information: full citation, abstract, references, citing

We describe a new technique for optimizing first-order functional programs. Program grammars, and optimization proceeds by counterexample: when a graph generates an unnecessary computation, the optimizer attempts to reformulate the grammar graph that contains that counterexample. This kind of program reformulation is for context-free grammars. Our reformulation ...

Keywords: functional languages, graph grammars, optimization

4 Editing by example

Robert P. Nix

October 1985 ACM Transactions on Programming Languages and Systems (TOPLA)

Full text available:  pdf(1.79 MB)

Additional Information: full citation, abstract, references, citing

An editing by example system is an automatic program synthesis facility embedded in a text editor to solve repetitive text editing problems. The user provides the editor with a few examples; the system analyzes the examples and generalizes them into a program that can perform the user's text. This paper presents an overview of the design, analysis, and implementation of the example system. It studies ...

5 Data-Driven and Demand-Driven Computer Architecture

Philip C. Treleaven, David R. Brownbridge, Richard P. Hopkins

January 1982 ACM Computing Surveys (CSUR), Volume 14 Issue 1

Full text available:  pdf(4.14 MB) Additional Information: full citation, references, citations, index terms

6 Automatic transformation of series expressions into loops

Richard C. Waters

January 1991 ACM Transactions on Programming Languages and Systems (TOPLA)

Full text available:  pdf(3.36 MB)

Additional Information: full citation, abstract, references, citing

The benefits of programming in a functional style are well known. In particular, compositions of functions operating on sequences/vectors/streams of data are more efficient than equivalent algorithms expressed as loops. Unfortunately, this kind of expression is not as it could be, for at least three reasons: (1) most programmers are less familiar with loops; (2) most programs ...

Keywords: sequences, series, streams, vectors

7 Toward a logical/physical theory of spreadsheet modeling

Tomás Isakowitz, Shimon Schocken, Henry C. Lucas

January 1995

ACM Transactions on Information Systems (TOIS), Volume 13

Full text available:  pdf(2.76 MB)

Additional Information: full citation, abstract, references, citin

In spite of the increasing sophistication and power of commercial spreadsheet methodology to support the construction and maintenance of spreadsheet mod perspective, we identify four principal components that characterize any sprea and binding. We present a factoring algorithm for identifying and extracting th


Keywords: model management

8 Editing by example

Robert Nix

January 1984

Proceedings of the 11th ACM SIGACT-SIGPLAN symposium on Princ

Full text available:  pdf(1.04 MB)

Additional Information: full citation, abstract, references, ci

An editing by example system is an automatic program synthesis facility embe solve repetitive text editing problems. The user provides the editor with a few system analyzes the examples and generalizes them into a program that can p the user's text. This paper presents the design, analysis, and implementation In particul ...

9 Software reuse

Charles W. Krueger

June 1992

ACM Computing Surveys (CSUR), Volume 24 Issue 2

Full text available:  pdf(4.96 MB)

Additional Information: full citation, abstract, references, c

Software reuse is the process of creating software systems from existing softw from scratch. This simple yet powerful vision was introduced in 1968. Softwar standard software engineering practice. In an attempt to understand why, rese software reuse and in the obstacles to implementing it. This paper surveys the found in the ...

Keywords: abstraction, cognitive distance, software reuse

10 Macro instruction extensions of compiler languages

M. Douglas McIlroy

April 1960

Communications of the ACM, Volume 3 Issue 4

Full text available:  pdf(831.69 KB)


Additional Information: full citation, abstract, refe


Macroinstruction compilers constructed from a small set of functions can be m conditional assembly, nested definitions, and parenthetical notation serve to m general extensions to its ground language.

- 11 A text-compression-based method for code size minimization in embedded**
Stan Liao, Srinivas Devadas, Kurt Keutzer
January 1999 ACM Transactions on Design Automation of Electronic Systems (TO
Full text available:  pdf(184.16 KB) Additional Information: full citation, abstract, references, citi

We address the problem of code-size minimization in VLSI systems with embe
reduces the production cost of embedded systemswe use data-compression m
strategies. In our framework, the compressed program consists of a skeleton a
dictionary can be computed by solving a set-covering problem derived from th
compressed code, we describe two me ...

Keywords: code size optimization, compression

- 12 Document Formatting Systems: Survey, Concepts, and Issues**
Richard Furuta, Jeffrey Scofield, Alan Shaw
September 1982 ACM Computing Surveys (CSUR), Volume 14 Issue 3
Full text available:  pdf(5.36 MB) Additional Information: full citation, references, citings, index terms

- 13 The <bigwig> project**
Claus Brabrand, Anders Møller, Michael I. Schwartzbach
May 2002 ACM Transactions on Internet Technology (TOIT), Volume 2 Issue
Full text available:  pdf(586.33 KB) Additional Information: full citation, abstract, referen

We present the results of the <bigwig> project, which aims to
domain-specific language for programming interactive Web ser

A fundamental aspect of the development of the World Wide Web di
change from static to dynamic generation of Web pages. Generating
with the client has the advantage of providing up-to-date and tailor-
of systems ...

Keywords: Interactive Web services, program analysis

14 Delegation logic: A logic-based approach to distributed authorization

Ninghui Li, Benjamin N. Grosz, Joan Feigenbaum

February 2003

ACM Transactions on Information and System Security (TISSEC)

Full text available:  pdf(316.24 KB)

Additional Information: full citation, abstract, referen

We address the problem of authorization in large-scale, open, distributed systems: electronic commerce, mobile-code execution, remote resource sharing, privacy. We adopt the trust-management approach, in which "authorization" is viewed as a set of credentials that prove that a request complies with a policy. We develop a *Logic* (DL), t ...


Keywords: Access control, Delegation Logic, distributed system security, logic

15 Extending attribute grammars to support programming-in-the-large

Josephine Micallef, Gail E. Kaiser

September 1994

ACM Transactions on Programming Languages and Systems (TO

Full text available:  pdf(2.76 MB)

Additional Information: full citation, abstract, references, i

Attribute grammars add specification of static semantic properties to context-free syntactic structure of program units. However, context-free grammars are not common in modern programming languages, including unordered collections of included units. We present extensions to context-free grammars, and corresponding suitable for defining such ...

Keywords: attribute evaluation, attribute grammars, include files, programming separate compilation, static semantics of programming languages

16 The CLP(R) language and system

Joxan Jaffar, Spiro Michaylov, Peter J. Stuckey, Roland H. C. Yap

May 1992

ACM Transactions on Programming Languages and Systems (TOPLAS)

Full text available:  pdf(3.73 MB)

Additional Information: full citation, abstract, references, cit

The CLP R programming language is defined, its underlying philosophy and important implementation issues are explored in detail, and finally, a prototype designed to be an instance of the Constraint Logic Programming Scheme ...

Keywords: constraints, logic programming

17 A theory of parameterized pattern matching: algorithms and applications

Brenda S. Baker

June 1993

Proceedings of the twenty-fifth annual ACM symposium on Theory of c

Full text available:  pdf(1.27 MB)

Additional Information: full citation, references, citations, index terms

18 A modal analysis of staged computation

Rowan Davies, Frank Pfenning

May 2001

Journal of the ACM (JACM), Volume 48 Issue 3

Full text available:  pdf(627.51 KB)

Additional Information: full citation, abstract, references

We show that a type system based on the intuitionistic modal logic $S4$ provide and analyzing computation stages in the context of typed λ -calculi and fun demonstrate the sense in which our $\lambda^{\text{arr}}_{\text{e}}$ -calculus captures staging, Nielson and Nielson's two-level functional language in our functional lan ...

Keywords: binding times, run-time code generation, staged computation

19 Code migration through transformations: an experience report

K. Kontogiannis, J. Martin, K. Wong, R. Gregory, H. Müller, J. Mylopoulos

November 1998

Proceedings of the 1998 conference of the Centre for Advanced S

Full text available:  pdf(203.77 KB)

Additional Information: full citation, abstract, refere

One approach to dealing with spiraling maintenance costs, manpower shortage is to "migrate" the code into a new platform and/or programming language. Th feasibility of semiautomating such a migration process in the presence of perfo migrant code. In particular, the paper reports on an experiment involving a m PL/IX. Several modules of the sy ...

20 Towards generic refactoring

Ralf Lämmel

October 2002

Proceedings of the 2002 ACM SIGPLAN workshop on Rule-based

Full text available:  pdf(135.66 KB)

Additional Information: full citation, abstract, refere

We define a challenging and meaningful benchmark for genericity in language *program refactoring*. We provide the first implementation of the benchmark ba Haskell. We use the basic refactoring of *abstraction extraction* as the running *functional programming framework* with hot spots for the language-specific ing

Keywords: frameworks, functional programming, generic programming, langua reuse, strafunski

Results 1 - 20 of 94

Result page: **1** 2 3 4 5 nex

The ACM Portal is published by the Association for Computing Machinery.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Con](#)

Useful downloads:  Adobe Acrobat  QuickTime  Windows Me